Utilizing Word Matching for Duplicate Article Removal: A Study using Malaysian Online News Feed

Tze-Wei Su, Hao-Ming Khor, Ian K T Tan
Faculty of Information Technology
Multimedia University
Cyberjaya 63100 Selangor, Malaysia
ian@mmu.edu.my

***Abstract*—Users of feed aggregators know that duplicated articles are found occasionally on the feeds they subscribe to. It can be time consuming to read all articles and stumble upon duplicated items they have already read. Our work here is to determine the effectiveness of using basic word matching to remove duplicated items and only show the most relevant item, thus saving readers' time. The method described in this paper to remove duplicates involves word matching heuristics with an appropriate matching percentage. The duplicated feeds are then ranked to only display the highest ranked article. Ranking is done using the number of search items found on the titles of the news feeds where the highest number returned will be considered the highest ranked article. Using Malaysian online news feeds, our method found that with a matching percentage of 40%, the method will be able to minimize duplicates.**

Keywords-component: duplicate removal, word matching, RSS feeds, news, ranking

I. INTRODUCTION

There are many online news feeds on the Internet where users can subscribe to. However, this often leads to having to read duplicated items which are unproductive. Online news feeds, such as The Star Online (www.thestar.com.my) and News Strait Times Online (nst.com.my), will definitely be reporting similar news daily. In most cases, they will have different titles for the same event and sometimes even the words in the titles can be different. This project aims to remove duplicated items and assists users to only read the most relevant articles.

Current feed aggregators do not have the ability to identify articles that are duplicates. We propose an efficient method to detect duplicates by using word matching where we compute the percentage of similar words used in different articles and through using a single matching percentage threshold, we identify them as duplicates or otherwise.

After identifying duplicated items, a further problem posed is which of the duplicated article should be shown to the user. In order to do so, we will need to rank the articles. We ranked the articles based on the number of search items returned using a search engine on the article's title. The higher the number of search items returned, the higher we will rank the article. The article's title with the most number of search returns will be used to display to the user.

Our contribution here is therefore;

- The identification of the appropriate word matching percentage for Malaysian online news feeds to identify duplicated articles.
- A news article ranking method using the number of search items returned on the articles' titles.

II. RELATED WORK

Two of the most common web-based feed aggregators are Drupal Aggregator Module [1] and Feed on Feeds [2]. Drupal [1] is a content management system (CMS) written in PHP. However, Drupal comes with a feed aggregator module with its default installation. Drupal's aggregator module can gather and parse feeds in RSS, Atom and RDF formats. It supports categorizing feeds and items can be filtered by keywords [1]. Drupal is not a standalone feed aggregator, and it does not have ranking nor duplicate removal features.

Feed on Feeds [2] is an open sourced PHP feed aggregator that requires a web server with PHP and MySQL installed. Feed on Feeds utilizes SimplePie's API [4] for aggregating feeds. There are several drawbacks that include the lack of sorting features and it does not have any duplication removal capability.

Most feed aggregators [2][3] are based on using available APIs that are widely available. In our work, we utilize SimplePie [4] which is a code library written in PHP to parse RSS and Atom feeds from websites. It has an API with various methods to extract data from feeds. SimplePie handles all the processing for fetching, caching and parsing the feeds. It has several weaknesses that include the fact that it does not use any database to store feeds and there are no functions to cater for ranking or duplication detection.

In the area of using words that are deemed statistically important, the article by Hirao et. al. [5] proposed the use of word statistcis to apply to news

articles. This combined with Clementine et. al. [6] can be used to identify titles that would contain similar contents. However, the computation requirements would be prohibitive and would not be suitable for a simple heuristics that we are looking for. This is generally the issue with most natural language processing solutions where computational overheads may be a hindrance for our intent.

Wang and Liu [7] proposed a duplication removal scheme for large scale web content that is based on the temporal vector of the article coupled with the feature codes of the article. Although this works well for large scale web content; usage of the temporal vector of an article is not relevant if we are considering duplicated news for the day, in other words, all contents of interest to us would be in the same period.

Takeda and Takasu [8] proposed content summary through the frequency of specific phrases (or "word sequences"). Our research uses this concept of matching words in order to determine duplication instead of summarization; with the intent to expand the work to also include weightage for word sequences.

In this work, we developed a prototype using the SimplePie API for fetching and extracting RSS data from the feeds the user subscribes. The API contains many useful functions that can be called by the system. SimplePie is used to extract data like titles, content and the date the articles were posted. All these data are obtained, stored in the database and shown to the users.

In obtaining duplicates, there is a need to determine the appropriate single news item that will be displayed. There are many proposals on ranking an article; such as work done by Svore et. al. [9] that uses learning neural networks combined with Wikipedia entries in order to determine the most important article. For simplicity, we loosely define ranking as search engine visibility where with more references to the article, it would indicate the importance of the content. As such, we will use the number of search engine returns as our ranking system where more returns would give higher ranking.

III. IMPLEMENTATION

The development of the prototype includes modules such as create accounts for users, login, add feeds, and view feed. We developed several complementary functions to assist our prototype development. The main function is the duplication removal, which we automated. If duplicates are found, it will then determine the ranking where it will select the highest ranked article and display it on the system interface. The lower ranked articles are not discarded but basically just hidden from view and users can still access them through an icon.

A. Word Matching

For the duplication section, we compare the words contained in the content of the articles. Titles are inappropriate to be used because titles for similar content may differ widely with no syntactic or semantic relevancy between the different similar articles.

The first phase in removing duplicates is to remove all punctuation symbols like commas, full stops, semicolons, and question marks. Then removal of all common words such as "the" and "an" are done. This is to increase efficiency and accuracy of the system as it does not need to compare too many words and also reduce the possibility of wrongly identifying duplicates because of the common words the articles have. Our database of common words were deduced from the Internet and used for this process. Once the data has been massaged, the number of word matching is conducted.

The words in one article are stored into one array and the words in the other article are stored in another array. We then used the PHP's built-in function (`array_intersect()`) to get a new array of words from both of the articles that matched. We then count the number of elements in this new array and divide it with the total number of words from both articles. After that, we calculate a percentage and identify duplicates through a percentage threshold. Only if duplication is detected would the articles be added to our database of duplicated articles. Below is a summary of our duplication method;

1. Initialize arrays to store articles to be compared and load common word database.
2. Pre-process the articles by:
   - Removal of HTML tags,
   - Converting all alphabetic content to lower case,
   - Removal of punctuations,
   - White space replacement, and
   - Removal of common words.
3. Remove duplicated words in within the arrays that store the articles.
4. Execute `array_intersect()` for both article arrays.
5. Execute `count_array_intersect()` to obtain the number of duplicates.
6. Compute using `match_percentage()` where the matching percentage is equal to $\frac{(\texttt{count\_array\_intersect()} \times 2) \times 100}{\texttt{(sum of both array count)}}$
7. If percentage is greater than threshold set, return match, else return no match.

B. *Ranking*

Ranking is the process of identifying which duplicated item is of higher importance and we use the highest ranked item to display to the user. Our method uses the title of the articles and obtains the number of search results. Titles are used as the content may contain too many search hits. This we did against the Microsoft Bing search engine. Upon obtaining a list of all duplicated articles, the titles of these duplicated items are searched using an automated script and the title that returns the highest results will be recorded as the highest ranked.

IV. RESULTS AND DISCUSSION

We did an empirical study using 5 news feeds. The 5 feeds were selected as they form the main stream news as well as alternative online news for Malaysia. We note that there are other online news feeds but we limited our initial studies to just the following;

- The Malaysian Insider
- News Straits Times
- The Star: Nation
- Google News: Malaysia
- Bernama

Data were collected over a period of 10 days and we recorded the number of items, number of duplicates found and manually determine the number of false positives as well as number of false negatives. False positives are articles that are wrongly identified as duplicates and false negatives are articles that the system did not manage to identify as duplicates. The testing was conducted over a range of word matching percentage (threshold values).

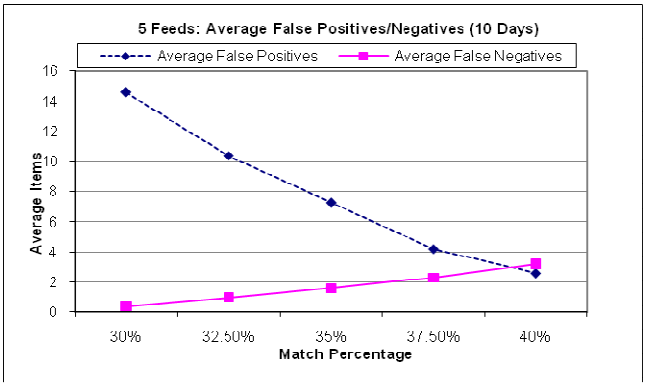

Figure 1. Average false positives / negatives for 5 feeds over a period of 10 days.

Figure 2 is the results from taking the average number of false positives and false negatives over a period of 10 days for all 5 news feeds. We can note that there is a compromise between trying to reduce the number of false positives with an increase in false negatives.

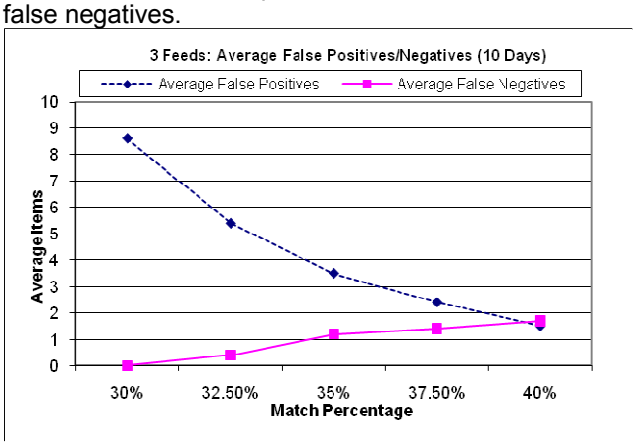

Figure 2. Average false positives / negatives for 3 feeds over a period of 10 days.

We tested the method using 3 news feeds instead of 5 and we found that the number of false positives and false negatives were reduced. This is depicted in figure 3. It is also noted that similarly to using 5 news feeds, the trade-off between reducing the number of false positives have an impact on the number of false negatives.

False positives have to be reduced significantly as we will not want the users to miss out on any news whilst false negatives are not as critical since the systems role is to help reduce the number of articles as oppose to completely eliminating duplicates. From the averaging of the number of false positives and false negatives indicates that there is a need to set the threshold to greater than 40% in order to minimize the number of false positives.

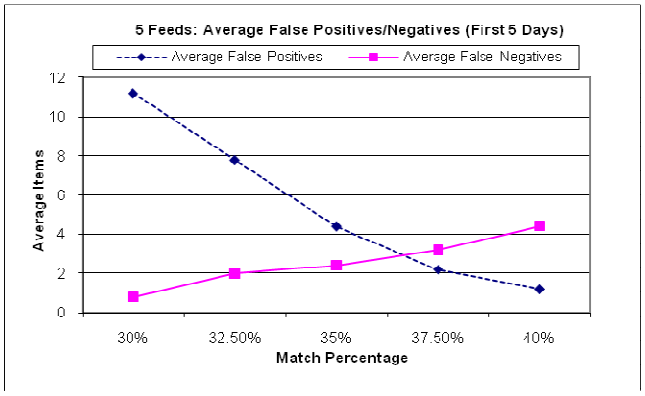

Figure 3. Average false positives / negatives for 5 feeds over the first 5 days.

We further analyzed our results by breaking them into first 5 days and next 5 days to determine the possible threshold value that is needed. Analyzing 5 feeds over a period of 5 days, it is noted (in figure 4) that the average number of false positives is less

than the number of false negatives at a lower threshold than over a span of 10 days. We then proceeded to analyze the results for the next 5 days.

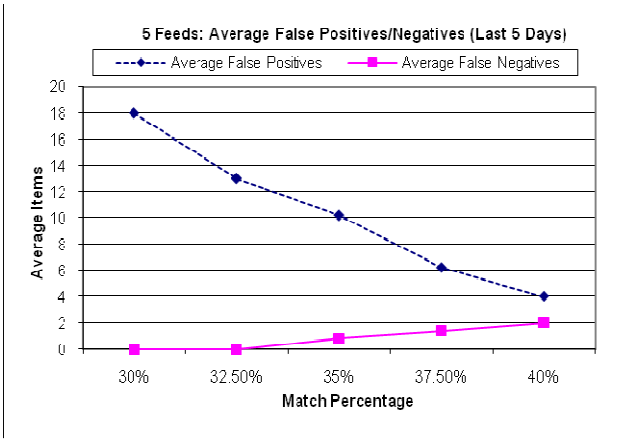

Figure 4. Average false positives / negatives for 5 feeds over the next 5 days.

Figure 5 illustrates the 2nd 5 days where the number of false positives are significant whilst the number of false negatives is reduced. From our findings with the limited data set that we have gathered, this method of determining the threshold values is at most a coarse estimation. To identify a useable match percentage threshold, we look for the point on the graph where the number of false positives and false negatives cross each other and select a threshold value to the right where there will be more false negatives than false positives.

Based on Figures 2 and 3, it indicates that a useable match percentage is greater than 40%. However, Figure 4 shows that it is about 36% while Figure 5 shows that it is more than 40%. We can conclude from this that because of the randomness of the articles for different days, it is difficult to know what should be the match percentage threshold that should be used. The fact that there are so many false positives in Figure 5 at 40% match threshold suggests that the news feeds contain many articles with the same event or person. Another determining factor is the number of news feeds. Naturally, the lesser number of feeds, this method will provide a better solution. This is because lesser news feeds mean lesser number of items to analyze, thus lesser chances of wrongly identifying duplicates. Although news feed items are random from day to day; the range of the match percentage as can be seen from the results should be within 38% to 39%.

V. Conclusion and Future Work

From our initial limited data set, the results gave a good indication that a basic word matching method can effectively achieve the task of removing duplicates with a careful selection of threshold values for the match percentage. It has shown to be effective in removing duplicates for Malaysia news feeds whilst being efficient on our prototype server. The authors note that the ranking method used is very primitive and the performance is highly dependent on external variables such as the Internet backbone, the number of articles being searched as well as the selected search engine's performance.

Our future work will entail a comprehensive set of data gathering using more news feeds and data gathering over a longer time period as opposed to just 10 days. There are strong indications that a higher threshold has to be set as more feeds are put into consideration but the authors are also conscious that typical news feed subscribers would generally not subscribe to too many sites.

This work can be used to eliminate false positives efficiently and coupled with other pre-processing heuristics; false negatives can be further minimized. The authors note that a simple word matching method will not be able to accurately remove duplication of news articles but it is an efficient method to reduce number of duplicates.

References

[1] *Aggregator: Publishing Syndicated Content* [Online]. (2009). Available: http://drupal.org/handbook/modules/aggregator [Accessed on 2010, November 30].

[2] S. Minutillo, *Welcome to Feed on Feeds, your server side, multi-user RSS and Atom aggregator!* (No date) [Online]. Available: http://feedonfeeds.com/ [Accessed on 2010, November 30].

[3] R. Parman and G. Sneddon, *What is SimplePie?* (2008). [Online]. Available: http://simplepie.org/wiki/faq/what_is_simplepie/ [Accessed on 2010, November 30].

[4] A. Rollett and A. Wood, *Rnews Feed Aggregator* (2009). [Online]. Available: http://rnews.sourceforge.net/ [Accessed on 2009, November 30].

[5] T. Hirao, M. Okumura, T. Fukushima, and H. Nanba. Text Summarization Challenge 3: Text Summarization Evaluation at NTCIR Workshop4. *In Proc. of the Fourth NTCIR Workshop on Research in Information Access Technologies: Information Retrieval, Question Answering, and Summarization. National Institute of Informatics*, 2004.

[6] C. Adam, E. Delpech, P. Saint-Dizier, *Identifying and Indexing Titles in Procedural Web Texts*, ACM International Conference on Document Engineering, Sao Paolo, 16 – 19 September 2008, 304-310.

[7] M. Wang, D. Liu, The Research of Web Page De-duplication Based on Web Pages Reshipment Statement, *In Proc. of First International Workshop on Database Technology and Applications*, Hubei, China, 25-26 April 2009.

[8] T. Takeda, A. Takasu, News Aggregating System with Automatic Summarization Based on Local Multiple Alignment, *In Proc. of The 6th International Conference on Informatics and Systems*, Cairo, Egypt, 27–29 March 2008. 65-73.

[9] K. M. Svore, L. Vanderwende, C. J. Burges, Enhancing Single-document Summarization by Combining RankNet and Third-party Sources, *In Proc. of The Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic, 28–30 June 2007. 448–457.